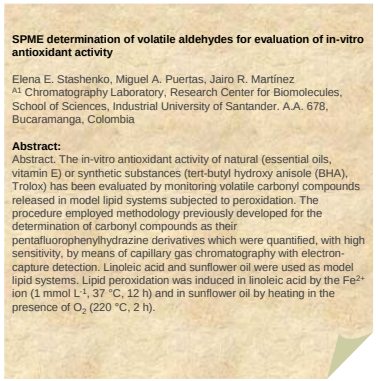
Document Model

The NLP Text Tools are document centric. That is, each of the tools, and their underlying java classes passes along or creates an instance of a Document. Each function or method adds content to this document instance. It is useful to understand the parts of the document class, how those parts interrelate, how those parts are assembled and by what process.

Document Model

Parts List

- Sections
- Sentences
- Phrases
- Terms
- Words
- Lexicon Entries

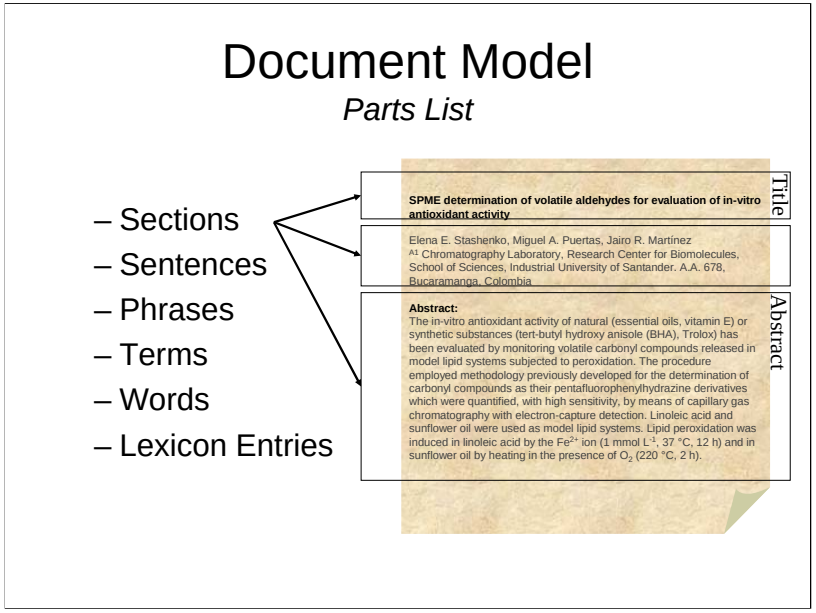


SPME determination of volatile aldehydes for evaluation of in-vitro antioxidant activity

Elena E. Stashenko, Miguel A. Puertas, Jairo R. Martínez
[A1] Chromatography Laboratory, Research Center for Biomolecules, School of Sciences, Industrial University of Santander. A.A. 678, Bucaramanga, Colombia

Abstract:
Abstract. The in-vitro antioxidant activity of natural (essential oils, vitamin E) or synthetic substances (tert-butyl hydroxy anisole (BHA), Trolox) has been evaluated by monitoring volatile carbonyl compounds released in model lipid systems subjected to peroxidation. The procedure employed methodology previously developed for the determination of carbonyl compounds as their pentafluorophenylhydrazine derivatives which were quantified, with high sensitivity, by means of capillary gas chromatography with electron-capture detection. Linoleic acid and sunflower oil were used as model lipid systems. Lipid peroxidation was induced in linoleic acid by the Fe^{2+} ion (1 mmol L^{-1}, 37 °C, 12 h) and in sunflower oil by heating in the presence of O_2 (220 °C, 2 h).

A high level parts list of the document includes sections, sentences, phrases, terms, words and entries from the SPECIALIST Lexicon.

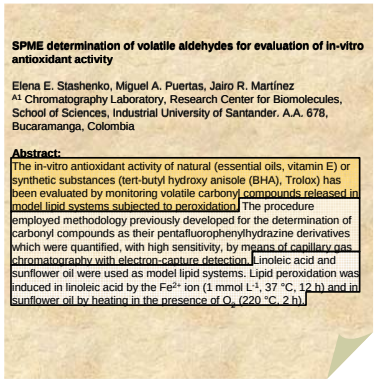


Sections are those explicitly noted components of a document such as the Title and Abstract sections of a MedLine Citation. When working with free text, where there are no explicitly annotated sections, each paragraph is equated with a section.

Document Model

Parts List

– Sections
– Sentences
– Phrases
– Terms
– Words
– Lexicon Entries



SPME determination of volatile aldehydes for evaluation of in-vitro antioxidant activity

Elena E. Stashenko, Miguel A. Puertas, Jairo R. Martínez
[A1] Chromatography Laboratory, Research Center for Biomolecules, School of Sciences, Industrial University of Santander. A.A. 678, Bucaramanga, Colombia

Abstract:

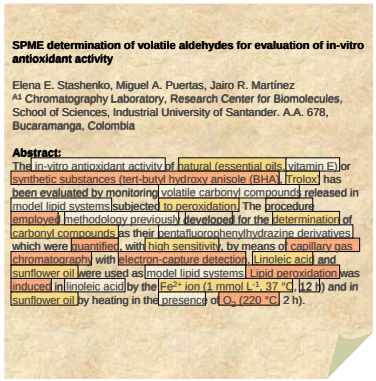
The in-vitro antioxidant activity of natural (essential oils, vitamin E) or synthetic substances (tert-butyl hydroxy anisole (BHA), Trolox) has been evaluated by monitoring volatile carbonyl compounds released in model lipid systems subjected to peroxidation. The procedure employed methodology previously developed for the determination of carbonyl compounds as their pentafluorophenylhydrazine derivatives which were quantified, with high sensitivity, by means of capillary gas chromatography with electron-capture detection. Linoleic acid and sunflower oil were used as model lipid systems. Lipid peroxidation was induced in linoleic acid by the Fe^{2+} ion (1 mmol L^{-1}, 37 °C, 12 h) and in sunflower oil by heating in the presence of O_2 (220 °C, 2 h).

We hope that we have the traditional notion of a sentence modeled here. For the post part, these are sequences of tokens that are bounded by sentence breaking punctuation such as periods, followed by patterns that indicate that a sentence has ended, typically one or two spaces followed by a string where the string has the first letter capitalized.

Document Model

Parts List

– Sections
– Sentences
– Phrases
– Terms
– Words
– Lexicon Entries



SPME determination of volatile aldehydes for evaluation of in-vitro antioxidant activity

Elena E. Stashenko, Miguel A. Puertas, Jairo R. Martínez
[a1] Chromatography Laboratory, Research Center for Biomolecules, School of Sciences, Industrial University of Santander. A.A. 678, Bucaramanga, Colombia

Abstract:
The in-vitro antioxidant activity of natural (essential oils, vitamin E) or synthetic substances (tert-butyl hydroxy anisole (BHA), Trolox) has been evaluated by monitoring volatile carbonyl compounds released in model lipid systems subjected to peroxidation. The procedure employed methodology previously developed for the determination of carbonyl compounds as their pentafluorophenylhydrazine derivatives, which were quantified, with high sensitivity, by means of capillary gas chromatography with electron-capture detection. Linoleic acid and sunflower oil were used as model lipid systems. Lipid peroxidation was induced in linoleic acid by the Fe^{2+} ion (1 mmol L^{-1}, 37 °C, 12 h) and in sunflower oil by heating in the presence of O_2 (220 °C, 2 h).

Phrases are at the heart of our document model. We want to do the bulk of our processing on phrases.

Document Model

Parts List

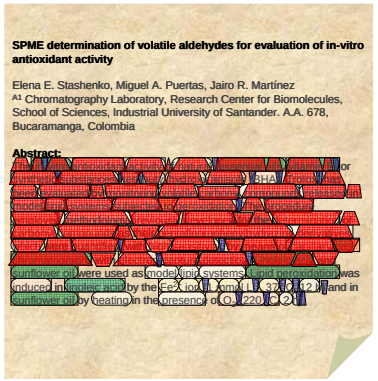
- Sections
- Sentences
- Phrases
- Terms
- Words
- Lexicon Entries

Terms are single or multi-word entities that hold some meaning on their own. The word “term” is so overloaded with meaning that we instead refer to instances of these parts as “lexical elements”.

Document Model

Parts List

– Sections
– Sentences
– Phrases
– Terms
– Words
– Lexicon Entries

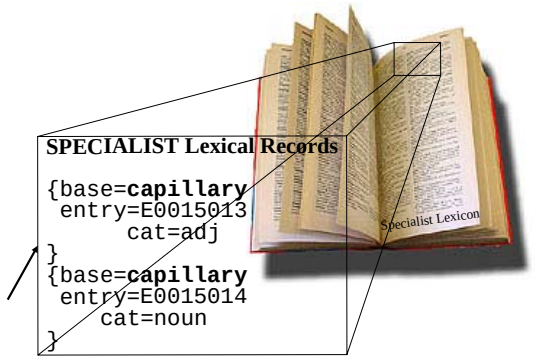


Words are also really useful to get at. The expression “word” is also so overloaded that we refer to instances of these parts as “tokens” A token for these tools is slightly different than the notion of a word for the lexical tools. A token here is a sequence of alpha numeric characters bounded by white space or punctuation or it is individual pieces of punctuation. The lexical tools notion of word removes punctuation, making it hard to put the original text back together again from the parts.

Document Model

Parts List

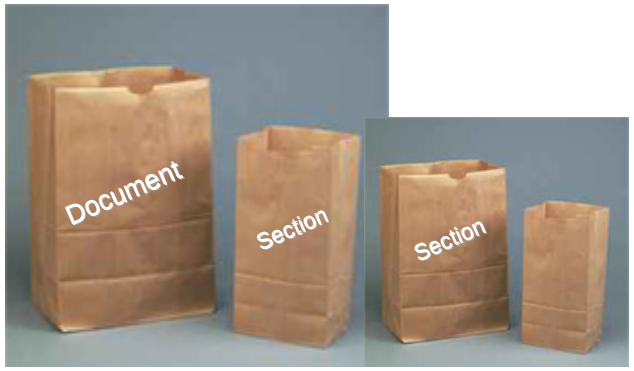
- Sections
- Sentences
- Phrases
- Terms
- Words
- Lexicon Entries



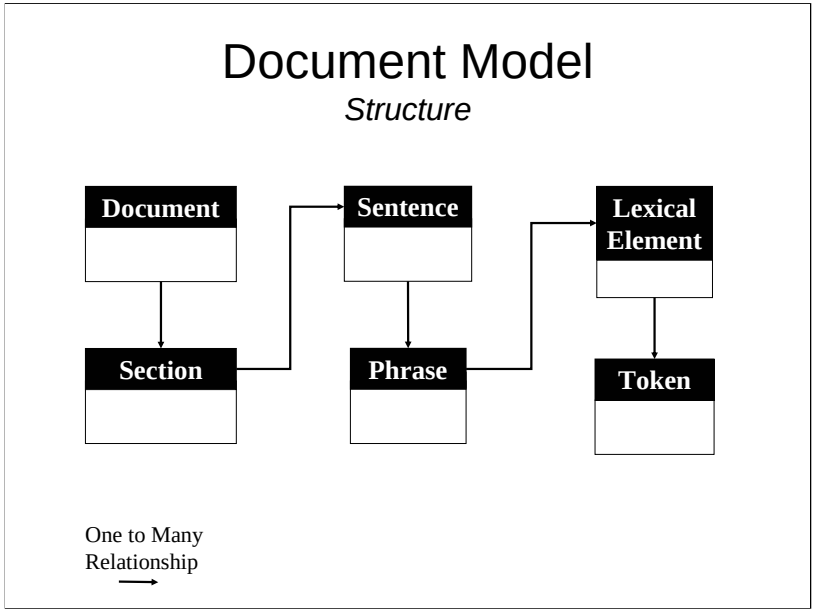
Lexicon Entries are entries from our SPECIALIST Lexicon. They are retrieved from the lexicon.

Document Model

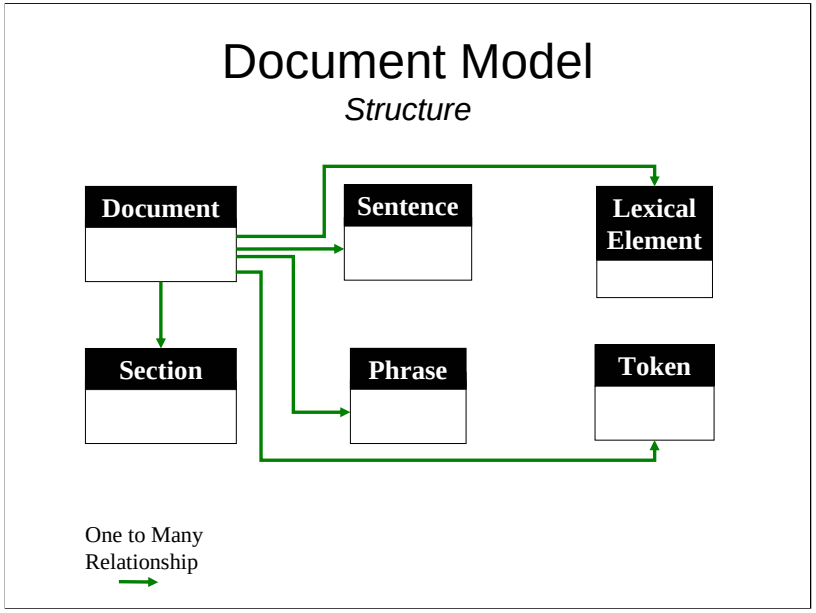
Structure



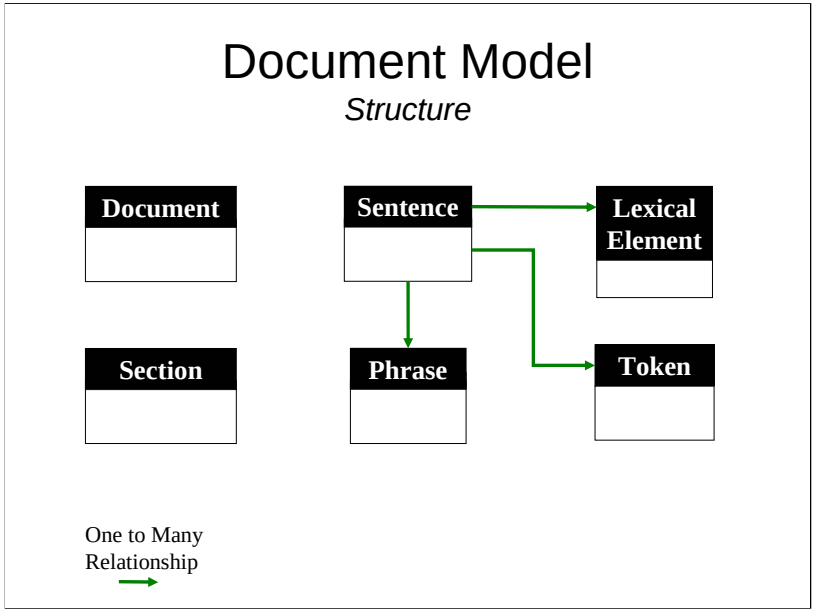
The parts build up structures that are useful to know about.



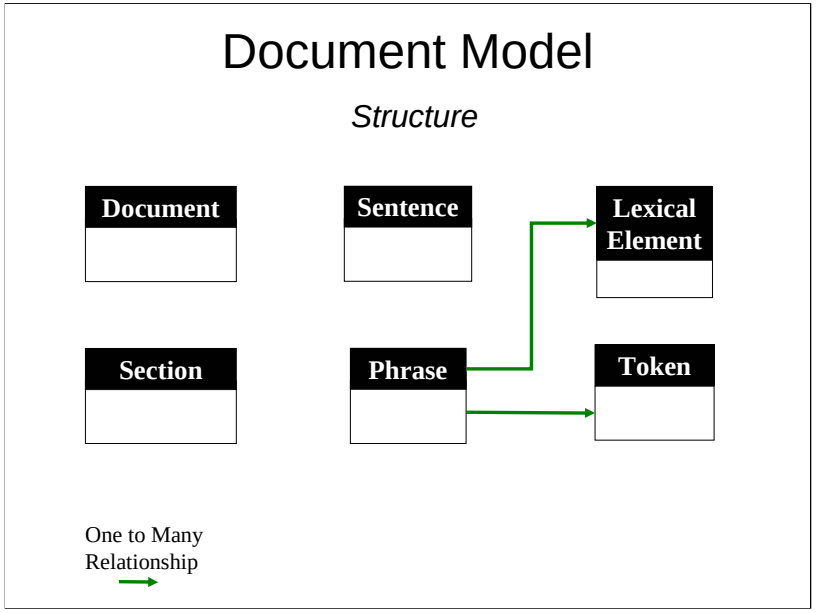
This is an entity diagram showing the primary relationship between each of the parts. One document contains one or more sections. Each section contains one or more sentences. Each sentence contains one or more phrases. Each phrase contains one or more lexical elements and each lexical element contains one or more tokens.



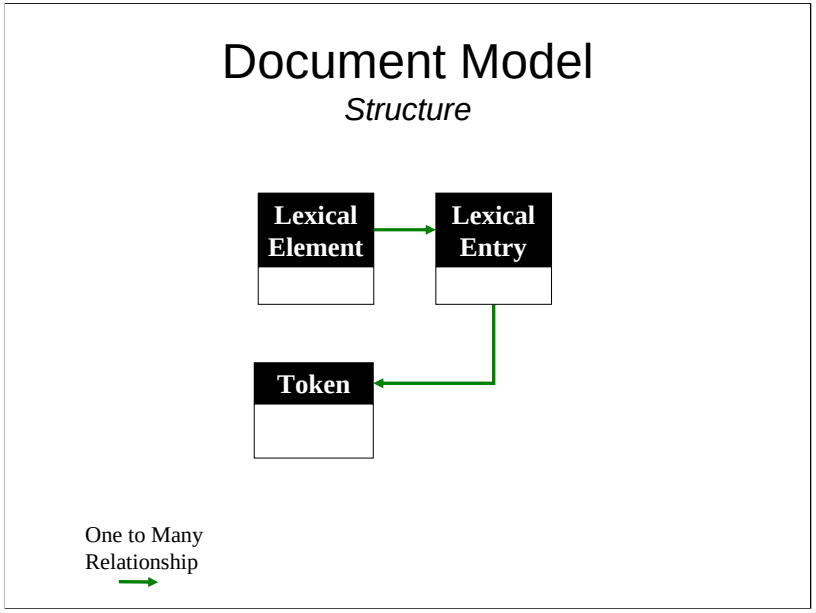
There are implicit relationships as well. A document contains not only one or more section, but also one or more sentence, one or more phrase, one or more lexical elements and one or more tokens. As a consequence, it would be correct to assume that there exists methods from the document class that retrieves sets of each of these other classes.



Just as a document has implicit relationships to the contained classes, the sentence object also contains one or more lexical elements, and one or more tokens. It would be correct in assuming that there are methods that can retrieve sets of these classes.



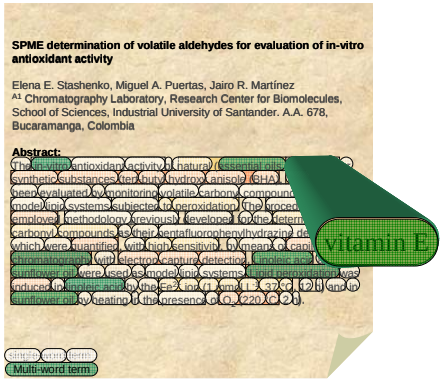
Phrases not only contain lexical elements, but also contain tokens that make up the lexical elements.



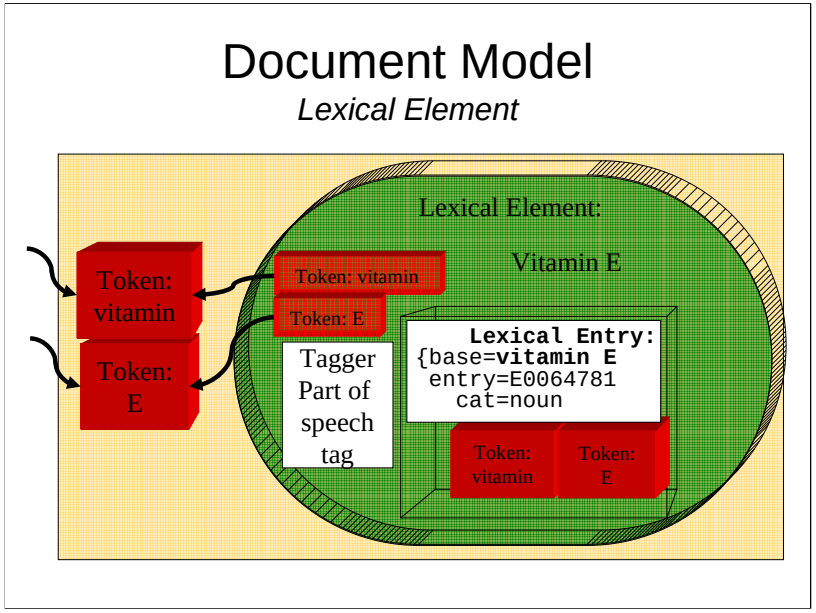
Lexical Elements contain lexical entries. These lexical entries refer to pieces not from the document, but pieces that come from the lexicon. Lexical entries, in tern, contain the tokens that make up lexical entries. The important point here is that the tokens that are contained in the lexical entry do not point to pieces from the document. Tokens normally contain character based offsets into the document that they came from, except for the tokens that come from lexical entries. It is meaningless to view the character based offsets of these tokens.

Document Model

Lexical Element



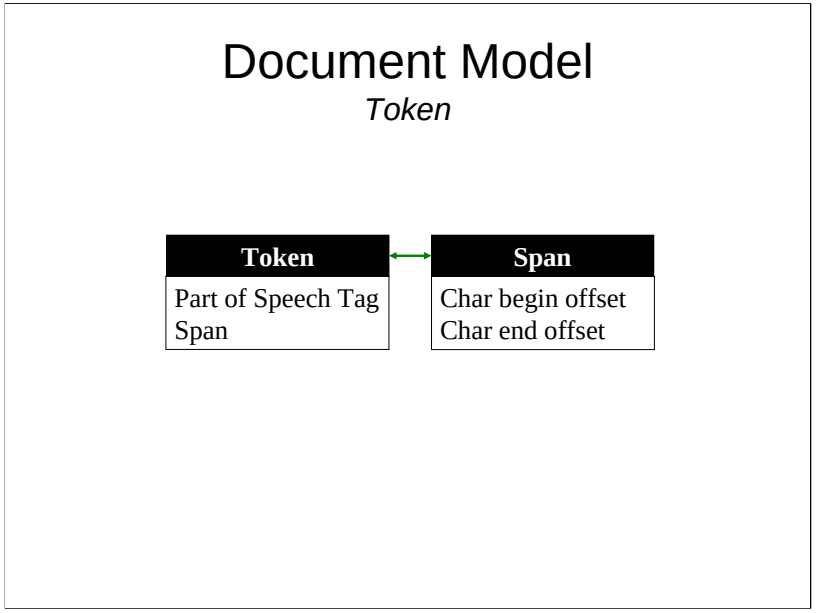
A lexical element in more detail. Take for example, the lexical element from this document which has the surface form “vitamin E”. Note that this is a multi-word lexical element.



The lexical element contains not only a copy of the original surface form “Vitamin E”, it also contains pointers to the tokens that make up this lexical element. These tokens are really pointers to the tokens that were initially tokenized by the tokenization process early on. It makes sense to have lexical elements refer to the tokens created during the tokenization process rather than have copies of them for several reasons. The first reason is to conserve memory space. The second reason is to note that other processes might be altering these tokens at some later time, adding additional information, say phrase offsets, or part of speech tags to these tokens. If copies of the tokens were added to the lexical elements, the resulting tokens would miss the updates to this token.

Lexical elements also contain the lexical entries that were found in the lexicon for this surface form. It may not be immediately obvious, but there could be multiple entries on a lexical element, because words can be ambiguous. For example, the word “sleep” could be either a noun or a verb. There would be an entry on the lexical element for each sense of the word. In the lexical element for “Vitamin E”, there is only one sense, and thus only one lexical entry.

As as noted before, Lexical Entries contain the tokens that make up the surface form that is found from the lexicon. These tokens are created when the lexical entry is created and what is on the lexical entry are the initial tokens. Note that although these tokens share the same surface forms, they are not the same instances as the tokens that point to the original document.



Tokens also contain a part of speech slot on them. This is a place holder for information that could come from a part of speech tagger that assigns part of speech tags for words rather than terms. The part of speech tags assigned from a tagger is propagated up to the lexical elements that contain these tokens. Rules are applied to resolve conflicts that occur between lexical elements that contain multiple tokens where the underlying tokens don't match the lexical element's part of speech tag.

Document Model

Phrase

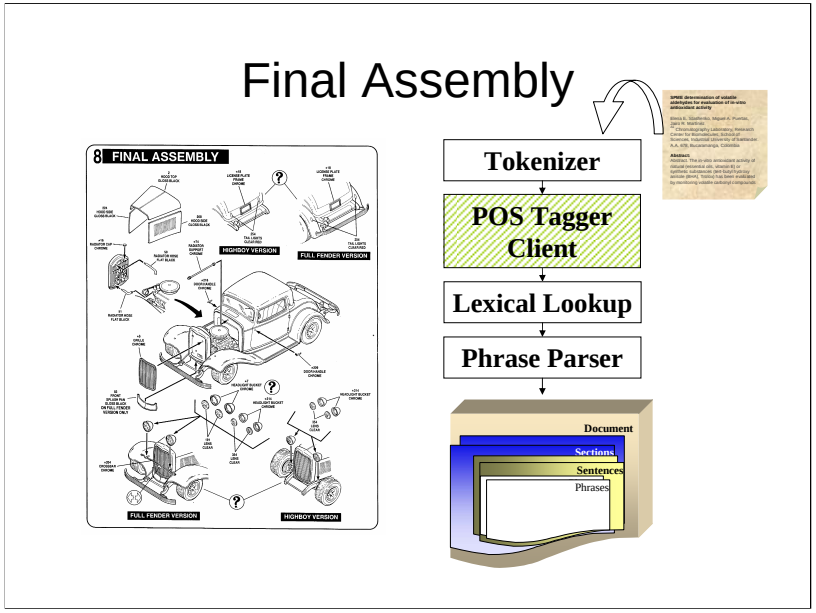
Phrase
String displayTags() String displayVariants() List getAllVariants() List getDerivedPhrases() List getLexicalElements() List getNp() String getNpString() List getNpTokens() String getOriginalString() int getPhrasePosition() int getSizeOfPhrase()

Phrase (cont.)
String getTrimmedString() boolean isOfPhrase() boolean isPrepPhrase() String toMincoManString() String toMoString() String toPipedString() String toString()

Since phrases are fundamental to text processing, here is a sample of some of the many methods associated with a phrase. Many of these are display methods, with methods to show only the essential parts of a phrase as in the getNp() methods. The essential parts of a phrase, which we sometimes refer to as the reduced noun phrase, are those parts of a phrase that would be found in a back-of-the-book kind of index. These are phrases that have had determiners, prepositions, verbs, and numbers removed. These are useful views of a phrase when using the phrase to retrieve from an index composed of a controlled vocabulary, such as Metathesaurus entries.

There are methods to return a partial bracketing or a shallow linguistic analysis of the phrase, noting the head or the important part of the phrase. This information is useful when comparing two phrases. Knowing whether both phrases contain the same head is a valuable criteria of how good a match is.

There are many, many more methods associated with the phrase. So many in fact, that it is likely that in a future iteration, this phrase class will be sub classed.



At the end of the phrase parsing, the final result is an instance of a document object that contains sections, sentences phrases, lexical elements and tokens. And those instances include methods that can retrieve useful information such as part of speech, character offsets, headedness.